CSCI 5440: Cryptography **Lecture 5**
The Chinese University of Hong Kong, Fall 2020 5 and 6 October 2020

Identification schemes are mechanisms for Alice to prove her identity to Bob. They comprise a setup phase in which some public or private information is shared between Alice and Bob and an identification protocol in which Alice's identity is established. The setup phase occurs, for example, when you register a password to a new website, or when you buy a new Octopus card. Identification occurs when you login or you swipe your Octopus.

In our study of encryption we were mainly concerned with passive adversaries. Eve could listen in to messages sent by Alice and Bob at will with the intent of learning forbidden information but she couldn't affect the communication.[1] To obtain reasonably secure identification we need to consider active adversaries—ones that can pretend to be Alice and/or Bob.

You are all familiar with password-based schemes. The secret information there is a password typically chosen by Alice. The identification consists of Alice sending her password or some function of it to Bob. One advantage of password-based schemes is that they are non-interactive; the protocol consists of just one message.

The security guarantees of passwords are quite weak. Because identification is deterministic, after observing a single interaction Eve can impersonate Alice. Eve can also steal Alice's password by impersonating Bob, namely engaging in a phishing attack. In typical scenarios Bob is a server that stores information about a large collection of passwords. If Eve gains unauthorized access to this server a lot of these passwords may be compromised. There are various methods for mitigating such attacks, some of them based on techniques that we will see in later lectures.

In this lecture we consider schemes that satisfy stronger notions of security. We begin with the secret-key setting. The protocols there are fairly straightforward, but this is a good playground to practice definitions.

1 Secret-key identification

In a secret-key identification scheme the setup phase consists of a common random choice of a secret key $K \sim \{0,1\}^k$ shared between Alice and Bob. It is standard terminology to call Alice *the Prover* and Bob *the Verifier*: Alice's job is to prove her identity to Bob, and Bob must verify this.

We model the prover and the verifier as (randomized) *interactive* circuits P and V. Each of them takes the secret key as input. The interaction consists of a sequence of messages exchanged between P and V. The protocol specifies the number of the messages, their direction,[2] as well as the length of each message. In the end the verifier produces a single output: accept if identification was successful, reject if not.

Definition 1. (P,V) is a *secret-key identification protocol* if upon interacting with $V(K)$, $P(K)$ accepts with probability 1.

A password protocol can be described as a one-message protocol: On input K, V sends the message K. The prover accepts if he received the message K and rejects otherwise.

Now consider an cheating prover P^* that wants to impersonate the honest prover Alice but without knowing the secret key. Whichever message P^* sends will be independent of the secret key K, so

[1]One exception is the chosen plaintext attack, in which Eve had control over the messages being encrypted.
[2]We can assume that directions alternate, so it is sufficient to specify the direction of the first message.

the prover will accept it with probability 2^{-k}. This is the chance of P^* guessing the secret key by chance. However, if P^* observes even a single interaction between Alice and Bob his chances of impersonating Alice jump to one.

This type of *eavesdropping attack* can be modeled by a two-phase game (see Figure **??** (a)). In the learning phase, the adversary observes some number q of independent transcripts[3] between $P(K)$ and $V(K)$. In the validation phase, the adversary plays a *cheating prover* P^**:* He must follow the format of the protocol from the prover's perspective, but can send messages of his choice. The *advantage* of the cheating prover in this game is the probability that the (honest) verifier $V(K)$ accepts this interaction.

Definition 2. (P, V) is (s, q, ε)-secure against eavesdropping if for every P^* of size at most s, after observing q interactions, the probability that P^* wins the eavesdropping game is at most ε.

The eavesdropping attack looks quite similar to the learning protocol from Lecture 2. The cheating prover is in fact trying to learn how to imitate an honest prover so he can pass validation. To prevent the learner from succeeding it is natural to use a pseudorandom function $F_K\colon \{0,1\}^n \to \{0,1\}^m$.

Challenge-response protocol:

1. Verifier chooses a random challenge $X \sim \{0,1\}^n$ and sends it to the verifier.
2. Prover responds with $Y = F_K(X)$.
3. Verifier accepts if Y equals $F_K(X)$.

Theorem 3. *If F is (s, q, ε)-hard to learn then the challenge-response protocol is $(s, q, \varepsilon + (q+1) \cdot 2^{-m})$-secure against eavesdropping attacks.*

By (s, q, ε)-hard to learn I mean that the advantage of the best q-query, size s-learner over randomly guessing the value $F_K(X)$ when tested on X is at most ε.

Proof. Suppose that some P^* wins the eavesdropping game with probability ε^*. To win the learning game, P^* must correctly predict $F_K(X)$ for some X that is different from all q that it has seen during the learning phase. Since the verifier issues independent challenges, the probability that this does not happen is at most $q \cdot 2^{-m}$. Therefore P^* wins the learning game with probability at least $\varepsilon^* - q \cdot 2^{-m}$. Its advantage over random of predicting $F_K(X)$ is $\varepsilon^* - (q+1) \cdot 2^{-m}$, so F cannot be $(s, q, \varepsilon^* - (q+1) \cdot 2^{-m})$-hard to learn. □

There is a more powerful type of attack in which the adversary actively impersonates the *verifier* (see Figure **??** (b)). This adversary can do more than just observe honest prover-verifier interactions; he controls the messages produced by the verifier. An *impersonation game* also consists of two phases. In the learning phase, the adversary plays a cheating verifier that participates in q interactions with the honest prover, sending messages of his choice. In the validation phase, the adversary can use the information learned from these interactions to identify himself to the honest verifier. The adversary's advantage is the probability that he is validated. Phishing scams in which a fake website V^* tries to steal users' credentials fit into this model.

The challenge-response protocol is secure even against impersonation attacks: Theorem **??** remains true if we replace the word "eavesdropping" by "impersonation". If an adversary controls the verifier's challenges in the learning phase, he cannot learn the pseudorandom function F_K with any higher advantage than what we showed in the proof of Theorem **??**.

[3]The *transcript* is the sequence of messages exchanged between P and V. When P or V take random inputs, like a shared key, the transcript is a random variable.

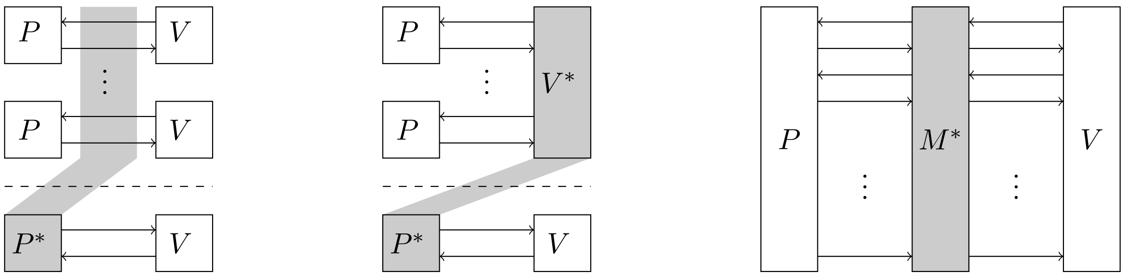

Figure 1: Identification attacks: (a) eavesdropping; (b) impersonation; (c) man-in-the-middle. The learning and validation phases are separated by the dashed line.

The most powerful adversary is a *man in the middle* who can alternate between playing prover and verifier roles throughout the execution of the protocol (see Figure **??** (c)). Identification that is secure against man-in-the-middle attacks is impossible unless additional assumptions are made about the underlying communication model.

2 Public-key identification

In public-key identification, during the setup phase the prover generates a key pair: A secret key SK that she gets to keep and a public key PK that is published to the verifier and any potential adversaries. The functionality requirement is that upon interacting with the verifier $V(PK)$, the prover $P(SK)$ accepts with probability one. Since the only distinguishing feature of the prover is her public key, a public-key identification protocol is a *proof of knowledge*: Alice has to prove to Bob that she knows her corresponding secret key.

Just like in the private-key setting, we can define eavesdropping and impersonation attacks. The difference here is that the adversary also knows the public key PK. To achieve this, our strategy will be to prevent the adversary from learning anything about the secret key. Since we don't know a public-key equivalent of pseudorandom function, we'll try to make use of its close relative, namely encryption. Let (Gen, Enc, Dec) be a public-key encryption scheme.

Public key challenge-response protocol:

1. Verifier chooses a random message $M \sim \{0,1\}^m$ and sends the encryption $C = Enc(PK, M)$.
2. Upon receiving C, Prover decrypts and responds with $M' = Dec(SK, M)$.
3. Verifier accepts if $M' = M$.

The protocol is clearly functional as long the underlying encryption scheme is. It is also secure against eavesdropping attacks: In the learning phase, an eavesdropper observes some encryptions and decryptions of various random messages. This is information that he can simulate on his own.

Claim 4. *If the encryption scheme is (s, ε)-message simulatability then the public-key challenge-response protocol is $(s - qt - O(m), q, \varepsilon + 2^{-m})$-secure against eavesdropping for every q, where t is the size of the encryption circuit.*

Proof. In each round of the learning phase, the eavesdropper observes encryptions of their messages and their decryptions, namely the messages themselves. In the validation phase, he gets a challenge $Enc(PK, M)$. The eavesdropper's view is the random variable

$$(PK, Enc(PK, M_1), M_1, \ldots, Enc(PK, M_q), M_q, Enc(PK, M)).$$

The pairs $(M_i, Enc(PK, M_i))$ can be sampled by running the encryption circuit, so they can be excluded from the eavesdropper's view at a cost of qt in size. Suppose this simplified eavesdropper A of size $s - O(m)$ can guess M with advantage better than desired, namely

$$\Pr[A(PK, Enc(PK, M)) = M] > \varepsilon + 2^{-m}.$$

Then A effectively decrypts random messages. This contradicts message simulatability: The output of every potential simulator Sim is independent of M, so

$$\Pr[A(PK, Sim(PK)) = M] \le 2^{-m}.$$

Therefore $(PK, Enc(PK, M), M)$ and $(PK, Sim(PK), M)$ are (s, ε)-distinguishable. (The $O(m)$ extra gates are for the equality.) This must hold for some fixed value of M, so $(PK, Enc(PK, M))$ and $(PK, Sim(PK))$ are also (s, ε) distinguishable for this M, contradicting simulatability. $\square$

Impersonation attacks are a different story. During the learning phase, the cheating verifier V^* has effective access to a *decryption oracle*: He can ask the prover to decrypt messages of his choice, not only random ones. We have not yet considered such adversaries against encryption.

To get a sense of what impersonation attacks on the challenge-response protocol can do, let us consider the case of El Gamal encryption. Recall that the key pair is $(SK = X, PK = g^X)$ and an encryption of M has the form $(g^R, g^{XR} \cdot M)$. The adversary can ask, for instance, for a decryption of the message $(PK, 1) = (g^X, 1)$. Since the only message that encrypts to this ciphertext is g^{-X^2}, the prover is obliged to produce this answer. So the adversary learns the power g^{X^2} of the *square* of the secret key; it is unclear how he could have calculated this value on his own without taking the discrete logarithm of the public key.

In conclusion, an adversary with access to a decryption oracle can potentially learn information that may be impossible to simulate efficiently. While it is not clear if this "knowledge" can help him mount an effective impersonation attack in this El Gamal-based challenge-response, it is not difficult to construct related examples in which challenge-response is insecure against impersonation even though the underlying encryption scheme is message-indistinguishable.

3 Schnorr's protocol

Schnorr proposed the following ingenious variation on El Gamal-based challenge-response. As usual $\mathbb{G}$ is a multiplicative group generated by g. The secret key is a random X from $\mathbb{Z}_q$ and the public key is g^X.

Schnorr's identification protocol:

1. Prover chooses a random $R \sim \mathbb{Z}_q$ (the commitment) and sends $h = g^R$ to the prover.
2. Verifier sends a random bit $C \sim \{0, 1\}$ to the prover.
3. Prover responds with $Y = R + CX$.
4. Verifier accepts if $g^Y = h \cdot PK^C$.

In a correct execution of the protocol, the verification equation is $g^{R+CX} = g^R \cdot (g^X)^C$. This is an identity so Schnorr's protocol is a legitimate identification protocol.

We will shortly look into the security of Schnorr's protocol against impersonation attacks. To gain some intuition let us first deal with the seemingly easier eavesdropping attacks. We'll assume that discrete logarithms base g are hard to calculate.

The view of the adversary consists of the public key $PK = g^X$, the observed transcript $(g^R, C, R + CX)$ for all queries in the learning phase and the messages received by the verifier in the validation phase.

Each observed interaction in the learning phase can be simulated by picking the prover's response before his commitment: The simulator $S(PK)$ chooses a random $C \sim \{0, 1\}$ and $Y \sim \mathbb{Z}_q$ and outputs $(g^Y \cdot PK^{-C}, C, Y)$. Since the joint distribution of $(C, R + CX)$ is uniform, (PK, C, Y) and $(PK, C, R + CX)$ are identically distributed. The remaining variables g^R and $g^Y \cdot PK^{-C}$ are deterministic functions of the rest and they are equal under the given change of variables. So the simulated transcript is identical to the original one. Just like in the proof of Claim **??**, we can omit the learning phase transcript from the eavesdropper's view at an additional cost of running this simulator q times.

In the validation phase, the cheating prover can choose any commitment h of his choice, not necessarily a random one. His commitment may even depend on the public key. Similarly, his response Y can be arbitrary. Yet, if he passes validation with advantage ε, for any choice of h and Y that he makes, it must be true that

$$\Pr[g^Y = h \cdot PK^C] \geq \varepsilon.$$

Now consider two correlated executions of the impersonation game that are identical up to step 2 (namely, the public key and the prover's first message are the same). In step 2, the verifier responds with two *independent* challenges C and C'. From here on, the two executions are (conditionally) independent. In step 3 the prover retorts with Y and Y', respectively. The probability that the verifier passes validation in both executions is

$$\begin{aligned}\Pr[g^Y = h \cdot PK^C \text{ and } g^{Y'} = h \cdot PK^{C'}] &= \mathrm{E}\big[\Pr_{C,C'}[g^Y = h \cdot PK^C \text{ and } g^{Y'} = h \cdot PK^{C'}]\big] \\ &= \mathrm{E}\big[\Pr_C[g^Y = h \cdot PK^C] \cdot \Pr_{C'}[g^{Y'} = h \cdot PK^{C'}]\big] \\ &= \mathrm{E}\big[\Pr_C[g^Y = h \cdot PK^C]^2\big].\end{aligned}$$

The first equation says that the probability that both executions validate can be calculated in two stages. First we fix all the randomness chosen before step two and then calculate the probability that from then on, for the random choices of C and C', both executions validate. Then we average over the remaining randomness.

The second equation says that because the events $g^Y = h \cdot PK^C$ and $g^Y = h \cdot PK^{C'}$ are independent conditioned on everything but the choice of C and C', the probability that they both occur is the product of their probabilities. The third equation says that these are the same because (C, Y) and (C', Y') are identically distributed.

The Cauchy-Schwarz inequality says that the average of a square is always at least as large as the square of an average, so

$$\Pr[g^Y = h \cdot PK^C \text{ and } g^{Y'} = h \cdot PK^{C'}] \geq \mathrm{E}\big[\Pr_C[g^Y = h \cdot PK^C]\big]^2 = \Pr[g^Y = h \cdot PK^C]^2 \geq \varepsilon^2.$$

In words, with probability ε^2 both of the following equations are satisfied:

$$\begin{aligned} g^Y &= h \cdot PK^C \\ g^{Y'} &= h \cdot PK^{C'}. \end{aligned} \tag{1}$$

Your natural instinct should lead you to solve this system of equations. You cancel h to obtain $g^{Y-Y'} = PK^{C'-C}$, and because the exponents form a group modulo a prime q, division in the exponent is allowed to get:

$$PK = g^{(Y-Y')/(C'-C)}.$$

In words, $(Y - Y')/(C' - C)$ is the discrete log of the public key. In summary, we applied the adversary on two correlated executions of Schnorr's protocol to calculate a discrete log. So the adversary cannot be efficient!

There is, unfortunately, an annoyance: C and C' are equal with probability half, in which case division is not allowed (and the two equations in (**??**) are the same). This is a symptom of a real problem: A cheating prover can, in fact, pass validation with probability 50%. This prover banks on the event that the challenge C is equal to zero. In this case, he does not need to know anything about the secret key. In step 3 he simply responds with R and is validated with flying colors!

Schnorr's protocol can be salvaged if the verifier's challenge C is chosen from a larger space. The security argument we just described remains valid even if C is uniformly random modulo q. The probability that C and C' are equal is then as small as $1/q$.

Schnorr's protocol with long challenges: Like Schnorr, except C is uniformly random in $\mathbb{Z}_q$.

Theorem 5. *If discrete logs in base g are (s, ε)-hard to find then Schnorr's protocol with long challenges is $(s/2 - t'/2 - q't, q', \sqrt{\varepsilon + 1/q})$-secure*[4] *against eavesdropping, where t is the size of the above simulator S, and t' is the size of two subtractions and one division in $\mathbb{G}$.*

Proof. By our discussion so far, if the eavesdropping game against Schnorr's protocol can be won with advantage ε^* by size s^* then there is an adversary of size $s^* + q't$ so that two correlated runs of this adversary's interaction with the verifier produce messages C, Y, C', Y' that satisfy the system (**??**). The value $(Y - Y')/(C' - C)$ equals the discrete log of g^X as long as C' and C are not equal, an event of probability $1/q$. So discrete logs can be computed with probability $\varepsilon = (\varepsilon^*)^2 - 1/q$ by size $s = 2(s^* + q't) + t'$. □

Like El Gamal-based challenge-response, Schnorr's protocol with long challenges is not known to be secure against impersonation attacks but is also not known to be insecure.

4 Simulation of cheating verifiers

To obtain security against impersonators we go back to the vanilla Schnorr protocol. An analogue of Theorem **??** still holds but the probability that the cheating prover passes the test can be bounded only by $\sqrt{1/2 + \varepsilon}$, which is rather high. We saw that it is rather easy for the prover to cheat with probability half.

A natural way to improve security is to repeat Schnorr's protocol independently many times. If the cheating prover can pass validation with probability, say, 3/4, we would expect that the chances that he passes the test t times should drop to $(3/4)^t$. This is indeed the case for eavesdropping attacks as long as the repetitions are carried out sequentially.

How about impersonation attacks? In the learning phase, the impersonator plays a cheating *verifier* V^*. In each round, V^* can choose the value of the challenge bit $C^* \in \{0, 1\}$ based on any previous information it has gained.

[4]We use q' for the number of rounds because q is taken for group size.

Let us start by looking at a single round of interaction between V^* and the honest prover. V^*'s view consists of the public key $PK = g^X$ and the pair of messages g^R and $R + C^*X$. This is summarized in the triple $(g^X, g^R, R + C^*X)$. The challenge C^* may not be random; V^* can bias it or even make it depend on g^X and g^R.

The objective is to simulate this view given the public key $PK = g^X$. Although the simulator has no control over the value of C^*, she still knows that V^*'s view is some combination of two random variables, each of them sampleable without knowing the secret key:

$$(g^X, g^R, R + C^*X) \text{ is distributed as } \begin{cases} (PK, g^Y, Y), & \text{if } C^* = 0 \\ (PK, g^Y \cdot PK^{-1}, Y), & \text{if } C^* = 1 \end{cases}$$

for a uniformly random $Y \sim \mathbb{Z}_q$. The simulator does not ahead of time what the value of C^* will be, but if she guesses it at random his guess will be correct with probability half. In the case that the simulator was incorrect, she can try again.

Simulator for Schnorr's protocol:

0. Choose a random bit $C \sim \{0, 1\}$ and a random Y from $\mathbb{Z}_q$.
1. Playing the role of the prover in Schnorr's protocol, send V^* the message $h = g^Y \cdot PK^{-C}$.
2. If V^* responds with C, output (PK, h, Y). If not, try again.

Conditioned on $C = C^*$, the output of the cut-choose simulator is identical to V^*'s view when interacting with the (honest) prover. Since h is independent of C and PK, C^* must be independent of C so C and C^* are equal with probability exactly $1/2$. The simulator therefore terminates within r rounds with probability $1 - 2^{-r}$.

Lemma 6. *The simulator terminated after at most r repetitions outputs a view that is $(\infty, 2^{-r})$-indistinguishable from V^* view when interacting with the honest prover.*

Lemma **??** generalizes to impersonation attacks with many rounds of interaction by sequentially composing the simulators. For a q'-round interaction the distinguishing probability drops to $q' \cdot 2^{-r}$. In conclusion, the impersonation attack against Schnorr's protocol has only a tiny advantage of $q' \cdot 2^{-r}$ over the eavesdropping attack. Schnorr's protocol, repeated sufficiently many times, remains secure against impersonation.